AFA
IFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of:

Sharon S. Liu, et al.

Serial No.: 09/483,724

Filed: January 14, 2000

For: MECHANISM FOR DYNAMICALLY CONSTRUCTING CUSTOMIZED IMPLEMENTATIONS TO ENFORCE RESTRICTIONS

Confirmation No.: 8756

Group Art Unit No.: 2131

Examiner: AKPATI, ODAICHE T.

MS Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed on October 6, 2004. This Appeal Brief conforms to the requirements of 37 CFR § 41.37(c).

I. REAL PARTY IN INTEREST

Sun Microsystems, Inc. is the real party in interest.

II. RELATED APPEALS AND INTERFERENCES

Appellant is unaware of any related appeals or interferences.

III. STATUS OF CLAIMS

Claims 2, 23, and 44 are canceled. Claims 1, 3-22, 24-43, and 45-66 are pending in this application, have been finally rejected, and are the subject of this appeal.

IV. STATUS OF AMENDMENTS

No amendments were filed after the Final Office Action mailed on April 7, 2004.

V. SUMMARY OF CLAIMED SUBJECT MATTER

With regard to Claim 1, there is recited a method performed by a framework in a system comprising the framework and at least one application, that method comprising:

> receiving a request from the application for a customized implementation of a service;
> determining a set of zero or more restrictions to be imposed upon said customized implementation;
> dynamically constructing said customized implementation, said customized implementation incorporating said restrictions, and comprising enforcement logic for enforcing said restrictions; and
> providing said customized implementation to the application;
> **wherein said customized implementation is invocable by the application without further interaction with the framework.**

With reference to Fig. 1, there is shown a block diagram of a system 100 in which the method of Claim 1 may be implemented. The system 100 comprises one or more applications 104, one or more general implementations 106, and a framework 102 for facilitating interaction between the various components. The applications 104 request and receive implementations of services from the framework 102. The term "service" is defined broadly to encompass any functionality requested by and provided to an application, including but not limited to encryption/decryption functionality (Application, page 5, lines 5-14).

When an application 104 requests an implementation from the framework 102, the application 104 specifies the type of service for which it wishes an implementation. For

example, the application 104 may request an implementation for the "Blowfish" encryption algorithm. In response, the framework 102 provides, to the application 104, an implementation of the requested service, which is customized for the application 104 making the request. The customized implementation provided by the framework 102 may contain restrictions on the services that it can provide (Application, page 5, lines 15-21).

The general implementations 106 represent the implementations for services that can be "plugged in" to or interfaced with the framework 102. Each general implementation 106 implements a particular type of service. For example, one general implementation may implement the "Blowfish" encryption algorithm, while another may implement the DES encryption algorithm. Each general implementation 106 is unrestricted. That is, the general implementations 106 themselves are not hampered by restrictions. In the case where the general implementations 106 are implementations of encryption algorithms, this means that the encryption algorithms may be set to full strength (Application, page 6, lines 1-10).

In system 100, the framework 102 is the component responsible for coordinating the overall operation of the system 100. A flowchart illustrating the general operation of the framework 102 is shown in Fig. 2. As shown in Fig. 2, the framework 102 operates by receiving (202) a request from an application 104 for an implementation of a particular type of service (e.g. an implementation for the "Blowfish" encryption algorithm). In response, the framework 102 determines (204) what restrictions, if any, need to be imposed on the requested implementation (Application, page 6, lines 13-19).

Once the restrictions are determined, the framework 102 dynamically constructs (206) the requested implementation. In one embodiment, the requested implementation is constructed by finding an associated general implementation 106 which implements the type of service

requested (e.g. a general implementation 106 which implements the "Blowfish" encryption algorithm). Once found, the associated general implementation 106 is incorporated into the requested implementation, along with the restrictions determined previously. In addition, a set of enforcement logic is also incorporated into the requested implementation. This enforcement logic ensures that the restrictions are enforced on the associated general implementation 106. Thus, even though the associated general implementation 106 itself may not have any restrictions, the enforcement logic causes the proper restrictions to be enforced on the associated general implementation 106. With the associated general implementation, the restrictions, and the enforcement logic incorporated therein, the construction of the requested implementation is complete. Since the requested implementation is constructed specifically for the requesting application 104, and hence, may incorporate restrictions specific to the requesting application, the requested implementation may be viewed as a customized implementation, which is customized for the requesting application 104 (Application, page 7, lines 1-18).

Once the customized implementation is constructed, it is provided (208) to the requesting application 104. Thereafter, the application 104 may directly request services from the customized implementation. Since the customized implementation incorporates the restrictions and enforcement logic for enforcing the restrictions, it is not necessary for the application 104 to further interact with the framework 102. The customized implementation itself will provide the services, and will guarantee that the restrictions are enforced on the services. By dynamically constructing customized implementations in this manner, the framework 102 ensures that the necessary restrictions are enforced on the services provided to the application 104 (Application, page 7, line 19, to page 8, line 2).

Claim 22 recites a system comprising at least one application and a framework comprising mechanisms that perform the steps of the method recited in Claim 1. Each mechanism is recited in means-plus-function form. An example of a mechanism for receiving a request from the application for a customized implementation of a service is shown in Fig. 3 as a GetInstance method of a Cipher object class 306 (Application, page 8, lines 17-19). An example of a mechanism for determining a set of zero or more restrictions to be imposed upon said customized implementation is shown in Fig. 3 as a GetCryptoPermission method of a JCESecurityManager class 316 (Application, page 12, lines 14-18). An example of a mechanism for dynamically constructing said customized implementation, said customized implementation incorporating said restrictions, and comprising enforcement logic for enforcing said restrictions, is shown in Fig. 3 as a constructor method of the Cipher object class 306 (Application, page 14, lines 12-16). An example of a mechanism for providing said customized implementation to the application is shown in Fig. 3 as the GetInstance method of the Cipher object class 306 (Application, page 15, lines 1-3). The application can invoke the methods of the customized implementation directly, without further interaction with the framework (Application, page 15, lines 3-4).

Claim 43 recites a computer-readable medium having stored thereon instructions which, when executed by one or more processors, cause the one or more processors to perform the steps of the method recited in Claim 1.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1, 3-16, 22, 24-37, 43, and 45-58 stand rejected under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,389,534 to Elgamal et al. ("Elgamal").

Claims 17-21, 38-42, and 59-63 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Elgamal in view of U.S. Patent No. 5,933,503 to Schell et al. ("Schell").

Claims 64-66 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Elgamal in view of U.S. Patent No. 6,005,942 to Chan et al. ("Chan").

VII. ARGUMENTS

A. The Limitations of Claims 1, 3-16, 22, 24-37, 43, and 45-58 Are Not in Any Way Taught or Suggested by Elgamal

Among other limitations, Claim 1 recites, "**wherein said customized implementation is invocable by the application without further interaction with the framework.**" As will be shown below, Elgamal does not teach or suggest a customized implementation that is invocable by an application without further interaction with a framework.

The method of Claim 1 is quite advantageous because it allows an application to obtain access to services without repeatedly requesting those services from some centralized framework. Specifically, when an application needs to access a particular service, it makes a request to a centralized framework for a customized implementation of that service. In response, the centralized framework dynamically constructs the customized implementation. The customized implementation includes enforcement logic for enforcing certain restrictions on the service. The centralized framework then provides the customized implementation to the requesting application. Notice that it is an invocable implementation of the service that is returned to the application, not a result of an operation. Because this is the case, the application can, in the future, obtain the service by invoking the customized implementation directly. The application does not need to interact with the framework again to obtain the service. By removing the framework from the service request/provision process, the method of Claim 1

removes the centralized framework as a potential performance bottleneck. This and other benefits can be derived from the method of Claim 1.

In order to determine whether Elgamal teaches or suggests a customized implementation that is invocable by an application without further interaction with the framework, it must be determined what component of Elgamal, if any, corresponds to the framework recited in Claim 1. Claim 1 makes it clear that the framework performs the step of receiving, from the application, a request for the customized implementation; thus, no component of Elgamal can qualify as such a framework unless that component receives such a request from an application.

The Examiner has not made it clear what component of Elgamal is supposed to correspond to the framework of Claim 1. However, in col. 5, lines 29-37, Elgamal discloses that applications 101-103 call service modules 104-107 to perform operations that perform cryptographic functions. Setting aside, for now, the question of whether such a call constitutes a "request for a customized implementation of a service," service modules 104-107 appear to be the only components of Elgamal that receive requests of any kind from applications 101-103. Therefore, in the absence of any contrary explanation from the Examiner, appellants can only assume that service modules 104-107 are supposed to correspond to part of the framework of Claim 1.

Having assumed this, it next must be determined whether Elgamal discloses any "customized implementation" that is invocable by at least one of applications 101-103 without further interaction (i.e., interaction that occurs after the returning of the customized implementation to the application) with service modules 104-107. Although the Examiner has not expressly stated what component of Elgamal is supposed to correspond to the customized implementation recited in Claim 1, the Examiner appears to allege that a combination of policy

filters 108-111 and crypto plugin modules 114-116 correspond to the customized implementation. When one of applications 101-103 is executed, crypto plugin modules 114-116 are loaded, and policy configuration module configures policy filters 108-111 (col. 5, lines 17-23). Apparently, the Examiner equates this configuration of policy filters 108-111 to the "dynamic construction" of a customized implementation required by Claim 1. When one of applications 101-103 calls one of service modules 104-107 to request an operation, that service module calls its corresponding policy filter to determine whether the called operation is allowed. If the corresponding policy filter does not approve the called operation, then the service module returns an error to the application. On the other hand, if the correspond policy filter approves the called operation, then the service module performs the requested operation, calling one of crypto plugin modules 114-116 as needed (col. 6, lines 32-43).

Notice that the applications 101-103 have to call the service modules 104-107 in order to get access to service. This is so even after the policy filters 108-111 have been configured and made available to the applications 101-103. Since the service modules 104-107 are part of the framework (as argued above), this means that the applications 101-103 always have to go through the framework to get access to a service. It is precisely this type of interaction that the method of claim 1 beneficially avoids. Unlike the method of claim 1, Elgamal does not provide a customized implementation to the applications 101-103 that can be invoked without further interaction with the framework.

Appellants previously submitted the above arguments to the Examiner in a response to the Final Office Action. In an Advisory Action mailed October 21, 2004, the Examiner responded that (a) the "cipher suites" (apparently referring to crypto plugin modules 114-116) are loaded once, (b) after this one-time loading operation, applications 101-103 interacted with

"other modules" (apparently referring to service modules 104-107) that approved or disapproved of "further processes" (apparently referring to application-requested operations) and (c) these interactions are not equivalent to the prior one-time loading process.

Although it is difficult to tell, the Examiner seems to be saying that the interaction of applications 101-103 with service modules 104-107 does not qualify as "further interaction with the framework" because the framework is something other than service modules 104-107. Once again, though, no component of Elgamal can qualify as the "framework" of Claim 1 unless, among other things, that component receives, from an application, a request for a customized implementation of a service. Elgamal does not identify the component that loads crypto plugin modules 114-116; Elgamal only appears to disclose that crypto plugin modules 114-116 "are loaded." Elgamal discloses that policy filter initialization module 112 configures policy filters 108-111. However, Elgamal does not appear to disclose that policy filter initialization module 112 ever receives a request from any of applications 101-103. Indeed, Elgamal does not appear to disclose that either the loading of crypto plugin modules 114-116 or the configuration of policy filters 108-111 in any way involves a request from one or more of applications 101-103; both the loading of crypto plugin modules 114-116 and the configuration of policy filters 108-111 is done in response to the execution of one of applications 101-103 rather than the receipt of a request from such an application (col. 5, lines 17-23).

As is stated above, service modules 104-107 appear to be the only components in Elgamal that ever receive a request from an application; it is only through service modules 104-107 that applications 101-103 request operations from crypto plugin modules 114-116. Thus, if service modules 104-107 do not correspond to the framework required by Claim 1, then Elgamal fails to teach or suggest a framework as required by Claim 1. Conversely, if service modules

104-107 correspond to the framework required by Claim 1, then applications 101-103 **necessarily** interact with the framework after the alleged dynamic construction of the alleged customized implementation and after the alleged provision of the alleged customized implementation to applications 101-103. However, the method of Claim 1 recites that the customized implementation is invocable by the application **without** such "further interaction." In either case, Elgamal fails to teach or suggest the method of Claim 1.

Based on the foregoing, it is therefore respectfully submitted that Elgamal does not in any way teach or suggest at least one of the limitations for which Elgamal was relied upon in the Final Office Action, namely, the limitation in Claim 1 of "**wherein said customized implementation is invocable by the application without further interaction with the framework.**" Claim 1 is therefore patentable over Elgamal.

Claims 3-16 depend from Claim 1 and include all of the limitations of Claim 1. It is therefore respectfully submitted that Claims 3-16 are patentable over Elgamal for at least the reasons set forth herein with respect to Claim 1.

Claims 22 and 24-37 contain limitations similar to Claims 1 and 3-16, respectively, except in the context of a framework. It is therefore respectfully submitted that Claims 22 and 24-37 are patentable over Elgamal for at least the reasons set forth herein with respect to Claims 1 and 3-16.

Claims 43 and 45-58 contain limitations similar to Claims 1 and 3-16, respectively, except in the context of a computer-readable medium. It is therefore respectfully submitted that Claims 43 and 45-58 are patentable over Elgamal for at least the reasons set forth herein with respect to Claims 1 and 3-16.

B. The Limitations of Claims 17-21, 38-42, and 59-63 Are Not in Any Way Taught or Suggested by Elgamal and Schell

By virtue of their dependence upon Claim 1, Claims 17-21 also contain the limitations of Claim 1, including the limitation of "wherein said customized implementation is invocable by the application without further interaction with the framework."

As is discussed above, Elgamal fails to teach or suggest at least this limitation. Thus, Claims 17-21 are patentable over Elgamal, taken individually.

Schell also fails to teach or suggest this limitation. Indeed, the Office Action relies only upon Elgamal to disclose this limitation. The Final Office Action does not even allege that Schell teaches or suggests this limitation. Thus, Claims 17-21 are patentable over Schell, taken individually.

Even assuming, arguendo, that it would have been obvious to combine Elgamal and Schell, the combination of Elgamal and Schell still fails to teach or suggest the limitation "wherein said customized implementation is invocable by the application without further interaction with the framework" as contained in Claims 17-21. It is therefore respectfully submitted that Claims 17-21 are patentable over Elgamal and Schell.

Claims 38-42 contain limitations similar to Claims 17-21, respectively, except in the context of a framework. It is therefore respectfully submitted that Claims 38-42 are patentable over Elgamal and Schell for at least the reasons set forth herein with respect to Claims 17-21.

Claims 59-63 contain limitations similar to Claims 17-21, respectively, except in the context of a computer-readable medium. It is therefore respectfully submitted that Claims 59-63 are patentable over Elgamal and Schell for at least the reasons set forth herein with respect to Claims 17-21.

C. The Limitations of Claims 64-66 Are Not in Any Way Taught or Suggested by Elgamal and Chan

By virtue of its dependence upon Claim 1, Claim 64 also contains the limitations of Claim 1, including the limitation of "wherein said customized implementation is invocable by the application without further interaction with the framework."

As is discussed above, Elgamal fails to teach or suggest at least this limitation. Thus, Claim 64 is patentable over Elgamal, taken individually.

Chan also fails to teach or suggest this limitation. Indeed, the Office Action relies only upon Elgamal to disclose this limitation. The Final Office Action does not even allege that Chan teaches or suggests this limitation. Thus, Claim 64 is patentable over Chan, taken individually.

Even assuming, arguendo, that it would have been obvious to combine Elgamal and Chan, the combination of Elgamal and Chan still fails to teach or suggest the limitation "wherein said customized implementation is invocable by the application without further interaction with the framework" as contained in Claim 64. It is therefore respectfully submitted that Claim 64 is patentable over Elgamal and Chan.

Additionally, the portion of Chan cited in the Final Office Action (col. 8, lines 30-44) refers to the "JAVA Card standard," rather than the "Java Cryptography Extension to Java Platform" recited in Claim 64. The "JAVA Card standard" **is not the same as** the "Java

Cryptography Extension to Java Platform" recited in Claims 64. Thus, Claim 64 is patentable over Chan, taken individually.

The Final Office Action admits that Elgamal does not teach or suggest a framework that comprises the Java Cryptography Extension to Java Platform as required by Claim 64. Thus, Claim 64 is patentable over Elgamal, taken individually.

Even assuming, arguendo, that it would have been obvious to combine Elgamal and Chan, the combination of Elgamal and Chan still fails to teach, or suggest a framework that comprises the Java Cryptography Extension to Java Platform as required by Claim 64. It is therefore respectfully submitted that Claim 64 is patentable over Elgamal and Chan.

Claim 65 and 66 contain limitations similar to Claim 64, except in the contexts of a system and a computer-readable medium, respectively. It is therefore respectfully submitted that Claims 65 and 66 are patentable over Elgamal and Chan for at least the reasons set forth herein with respect to Claims 64.

VIII. CONCLUSION AND PRAYER FOR RELIEF

Based on the foregoing, it is respectfully submitted the rejections of Claims 1, 3-16, 22, 24-37, 43, and 45-58 lack the requisite factual and legal bases. Appellants therefore respectfully request that the Honorable Board reverse the rejections of Claims 1, 3-16, 22, 24-37, 43, and 45-58.

Respectfully submitted,

HICKMAN PALERMO TRUONG & BECKER LLP

Date: December 6, 2004

Christian A. Nicholes
Registration No. 50,266

1600 Willow Street
San Jose, California 95125-5106
Tel: (408) 414-1224
Fax: (408) 414-1076

CERTIFICATE OF MAILING

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: MS Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450

on 12/6/04 by Judy Paradoski

Judy Paradoski

CLAIMS APPENDIX

1. In a system comprising at least one application and a framework, a method performed by the framework comprising:

receiving a request from the application for a customized implementation of a service;

determining a set of zero or more restrictions to be imposed upon said customized implementation;

dynamically constructing said customized implementation, said customized implementation incorporating said restrictions, and comprising enforcement logic for enforcing said restrictions; and

providing said customized implementation to the application;

wherein said customized implementation is invocable by the application without further interaction with the framework.

3. The method of claim 1, wherein the system further comprises a general implementation for said service, wherein said general implementation is unrestricted, and wherein said customized implementation further incorporates said general implementation.

4. The method of claim 3, wherein said enforcement logic enforces said restrictions on said general implementation.

5. The method of claim 1, wherein said enforcement logic is invoked upon initialization of said customized implementation.

6. The method of claim 5, wherein said enforcement logic, when invoked:

receives a set of desired parameters from the application;

determines whether the desired parameters exceed said restrictions; and

in response to a determination that the desired parameters exceed said restrictions, preventing said customized implementation from operating.

7. The method of claim 5, wherein said service is an encryption/decryption service, and wherein said enforcement logic, when invoked:

determines whether a particular exemption mechanism has been invoked; and

in response to a determination that the particular exemption mechanism has not been invoked, preventing said customized implementation from operating.

8. The method of claim 1, wherein determining the set of zero or more restrictions comprises:

accessing information specifying one or more limitations; and

processing said limitations to derive said restrictions.

9. The method of claim 8, wherein said service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations.

10. The method of claim 9, wherein said default encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

11. The method of claim 1, wherein determining the set of zero or more restrictions comprises:

accessing information specifying one or more limitations;

determining permissions, if any, granted to the application; and

reconciling said limitations and said permissions to derive said restrictions.

12. The method of claim 11, wherein said limitations and said permissions are reconciled to derive restrictions which are least restrictive.

13. The method of claim 11, wherein said service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations, and a set of zero or more exempt encryption limitations which apply when one or more exemption mechanisms are implemented.

14. The method of claim 13, wherein said default encryption limitations and said exempt encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

15. The method of claim 13, wherein reconciling said limitations and said permissions comprises:

determining whether the application has been granted any permissions; and

in response to a determination that the application has not been granted any permissions, deriving said restrictions from said set of default encryption limitations.

16. The method of claim 13, wherein reconciling said limitations and said permissions comprises:

determining whether the application has been granted any permissions which require implementation of a particular exemption mechanism;

in response to a determination that the application has been granted a permission which requires implementation of a particular exemption mechanism, determining whether said exempt encryption limitations allow said particular exemption mechanism to be implemented; and

in response to a determination that said exempt encryption limitations allow said particular exemption mechanism to be implemented, deriving said restrictions from said set of exempt encryption limitations.

17. The method of claim 1, wherein the system further comprises a general implementation for said service, and wherein dynamically constructing said customized implementation comprises:

instantiating the general implementation to give rise to a general implementation instance;

instantiating a wrapper object; and

encapsulating said general implementation instance and said restrictions within said wrapper object to derive said customized implementation.

18. The method of claim 17, wherein said wrapper object comprises one or more invocable methods, wherein said general implementation instance comprises one or more invocable methods, and wherein encapsulating comprises:

mapping one or more of the invocable methods of said wrapper object to one or more of the invocable methods of said general implementation instance.

19. The method of claim 18, wherein said wrapper object comprises initialization logic for enforcing said restrictions on said general implementation instance.

20. The method of claim 19, wherein said initialization logic is invoked prior to allowing any of the invocable methods of said general implementation instance to be invoked.

21. The method of claim 17, further comprising:

instantiating an exemption mechanism to give rise to an exemption mechanism instance; and

encapsulating said exemption mechanism instance within said wrapper object.

22. In a system comprising at least one application, a framework comprising:

a mechanism for receiving a request from the application for a customized implementation of a service;

a mechanism for determining a set of zero or more restrictions to be imposed upon said customized implementation;

a mechanism for dynamically constructing said customized implementation, said customized implementation incorporating said restrictions, and comprising enforcement logic for enforcing said restrictions; and

a mechanism for providing said customized implementation to the application;

wherein said customized implementation is invocable by the application without further interaction with the framework.

24. The framework of claim 22, wherein the system further comprises a general implementation for said service, wherein said general implementation is unrestricted, and wherein the mechanism for dynamically constructing said customized implementation further incorporates said general implementation within said customized implementation.

25. The framework of claim 24, wherein said enforcement logic enforces said restrictions on said general implementation.

26. The framework of claim 22, wherein said enforcement logic is invoked upon initialization of said customized implementation.

27. The framework of claim 26, wherein said enforcement logic, when invoked:

receives a set of desired parameters from the application;

determines whether the desired parameters exceed said restrictions; and

in response to a determination that the desired parameters exceed said restrictions, preventing said customized implementation from operating.

28. The framework of claim 26, wherein said service is an encryption/decryption service, and wherein said enforcement logic, when invoked:

determines whether a particular exemption mechanism has been invoked; and

in response to a determination that the particular exemption mechanism has not been invoked, preventing said customized implementation from operating.

29. The framework of claim 22, wherein the mechanism for determining the set of zero or more restrictions comprises:

a mechanism for accessing information specifying one or more limitations; and

a mechanism for processing said limitations to derive said restrictions.

30. The framework of claim 29, wherein said service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations.

31. The framework of claim 30, wherein said default encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

32. The framework of claim 22, wherein the mechanism for determining the set of zero or more restrictions comprises:

a mechanism for accessing information specifying one or more limitations;

a mechanism for determining permissions, if any, granted to the application; and

a mechanism for reconciling said limitations and said permissions to derive said restrictions.

33. The framework of claim 32, wherein said limitations and said permissions are reconciled to derive restrictions which are least restrictive.

34. The framework of claim 32, wherein said service is an encryption/decryption service, and wherein said information comprises a set of one or

more default encryption limitations, and a set of zero or more exempt encryption limitations which apply when one or more exemption mechanisms are implemented.

35. The framework of claim 34, wherein said default encryption limitations and said exempt encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

36. The framework of claim 34, wherein the mechanism for reconciling said limitations and said permissions comprises:

a mechanism for determining whether the application has been granted any permissions; and

a mechanism for deriving, in response to a determination that the application has not been granted any permissions, said restrictions from said set of default encryption limitations.

37. The framework of claim 34, wherein the mechanism for reconciling said limitations and said permissions comprises:

a mechanism for determining whether the application has been granted any permissions which require implementation of a particular exemption mechanism;

a mechanism for determining, in response to a determination that the application has been granted a permission which requires implementation of a particular exemption mechanism, whether said exempt encryption limitations allow said particular exemption mechanism to be implemented; and

a mechanism for deriving, in response to a determination that said exempt encryption limitations allow said particular exemption mechanism to be implemented, said restrictions from said set of exempt encryption limitations.

38. The framework of claim 22, wherein the system further comprises a general implementation for said service, and wherein the mechanism for dynamically constructing said customized implementation comprises:

a mechanism for instantiating the general implementation to give rise to a general implementation instance;

a mechanism for instantiating a wrapper object; and

a mechanism for encapsulating said general implementation instance and said restrictions within said wrapper object to derive said customized implementation.

39. The framework of claim 38, wherein said wrapper object comprises one or more invocable methods, wherein said general implementation instance comprises one or more invocable methods, and wherein the mechanism for encapsulating comprises:

a mechanism for mapping one or more of the invocable methods of said wrapper object to one or more of the invocable methods of said general implementation instance.

40. The framework of claim 39, wherein said wrapper object comprises initialization logic for enforcing said restrictions on said general implementation instance.

41. The framework of claim 40, wherein said initialization logic is invoked prior to allowing any of the invocable methods of said general implementation instance to be invoked.

42. The framework of claim 38, further comprising:

a mechanism for instantiating an exemption mechanism to give rise to an exemption mechanism instance; and

a mechanism for encapsulating said exemption mechanism instance within said wrapper object.

43. In a system comprising at least one application, a computer readable medium having stored thereon instructions which, when executed by one or more processors, cause the one or more processors to implement a framework which dynamically constructs a customized implementation of a service, said computer readable medium comprising:

instructions for causing one or more processors to receive a request from the application for a customized implementation of a service;

instructions for causing one or more processors to determine a set of zero or more restrictions to be imposed upon said customized implementation;

instructions for causing one or more processors to dynamically construct said customized implementation, said customized implementation incorporating said restrictions, and comprising enforcement logic for enforcing said restrictions; and

instructions for causing one or more processors to provide said customized implementation to the application;

wherein said customized implementation is invocable by the application without further interaction with the framework.

45. The computer readable medium of claim 43, wherein the system further comprises a general implementation for said service, wherein said general implementation is unrestricted, and wherein said customized implementation further incorporates said general implementation.

46. The computer readable medium of claim 45, wherein said enforcement logic enforces said restrictions on said general implementation.

47. The computer readable medium of claim 43, wherein said enforcement logic is invoked upon initialization of said customized implementation.

48. The computer readable medium of claim 47, wherein said enforcement logic, when invoked:

receives a set of desired parameters from the application;

determines whether the desired parameters exceed said restrictions; and

in response to a determination that the desired parameters exceed said restrictions, preventing said customized implementation from operating.

49. The computer readable medium of claim 47, wherein said service is an encryption/decryption service, and wherein said enforcement logic, when invoked:

determines whether a particular exemption mechanism has been invoked; and

in response to a determination that the particular exemption mechanism has not been invoked, preventing said customized implementation from operating.

50. The computer readable medium of claim 43, wherein the instructions for causing one or more processors to determine the set of zero or more restrictions comprises:

instructions for causing one or more processors to access information specifying one or more limitations; and

instructions for causing one or more processors to process said limitations to derive said restrictions.

51. The computer readable medium of claim 50, wherein said service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations.

52. The computer readable medium of claim 51, wherein said default encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

53. The computer readable medium of claim 43, wherein the instructions for causing one or more processors to determine the set of zero or more restrictions comprises:

instructions for causing one or more processors to access information specifying one or more limitations;

instructions for causing one or more processors to determine permissions, if any, granted to the application; and

instructions for causing one or more processors to reconcile said limitations and said permissions to derive said restrictions.

54. The computer readable medium of claim 53, wherein said limitations and said permissions are reconciled to derive restrictions which are least restrictive.

55. The computer readable medium of claim 53, wherein said service is an encryption/decryption service, and wherein said information comprises a set of one or more default encryption limitations, and a set of zero or more exempt encryption limitations which apply when one or more exemption mechanisms are implemented.

56. The computer readable medium of claim 55, wherein said default encryption limitations and said exempt encryption limitations are derived by merging multiple jurisdiction policies and extracting therefrom the most restrictive encryption limitations.

57. The computer readable medium of claim 55, wherein the instructions for causing one or more processors to reconcile said limitations and said permissions comprises:

instructions for causing one or more processors to determine whether the application has been granted any permissions; and

instructions for causing one or more processors to derive, in response to a determination that the application has not been granted any permissions, said restrictions from said set of default encryption limitations.

58. The computer readable medium of claim 55, wherein the instructions for causing one or more processors to reconcile said limitations and said permissions comprises:

instructions for causing one or more processors to determine whether the application has been granted any permissions which require implementation of a particular exemption mechanism;

instructions for causing one or more processors to determine, in response to a determination that the application has been granted a permission which requires implementation of a particular exemption mechanism, whether said exempt encryption limitations allow said particular exemption mechanism to be implemented; and

instructions for causing one or more processors to derive, in response to a determination that said exempt encryption limitations allow said particular exemption mechanism to be implemented, said restrictions from said set of exempt encryption limitations.

59. The computer readable medium of claim 43, wherein the system further comprises a general implementation for said service, and wherein the instructions for causing one or more processors to dynamically construct said customized implementation comprises:

instructions for causing one or more processors to instantiate the general implementation to give rise to a general implementation instance;

instructions for causing one or more processors to instantiate a wrapper object; and

instructions for causing one or more processors to encapsulate said general implementation instance and said restrictions within said wrapper object to derive said customized implementation.

60. The computer readable medium of claim 59, wherein said wrapper object comprises one or more invocable methods, wherein said general implementation instance comprises one or more invocable methods, and wherein the instructions for causing one or more processors to encapsulate comprises:

instructions for causing one or more processors to map one or more of the invocable methods of said wrapper object to one or more of the invocable methods of said general implementation instance.

61. The computer readable medium of claim 60, wherein said wrapper object comprises initialization logic for enforcing said restrictions on said general implementation instance.

62. The computer readable medium of claim 61, wherein said initialization logic is invoked prior to allowing any of the invocable methods of said general implementation instance to be invoked.

63. The computer readable medium of claim 59, further comprising:

instructions for causing one or more processors to instantiate an exemption mechanism to give rise to an exemption mechanism instance; and

instructions for causing one or more processors to encapsulate said exemption mechanism instance within said wrapper object.

64. The method of claim 1, wherein said framework comprises Java Cryptography Extension to Java Platform.

65. The framework of claim 22, wherein said framework comprises Java Cryptography Extension to Java Platform.

66. The computer readable medium of claim 43, wherein said framework comprises Java Cryptography Extension to Java Platform.

PTO/SB/17 (12/99)
Approved for use through 09/30/2000. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE

FEE TRANSMITTAL for FY 2005

Patent fees are subject to annual revision,
Small Entity payments must be supported by a small entity statement, otherwise large entity fees must be paid. See Forms PTO/SB/09-12.
See 37 C.F.R. §§ 1.27 AND 1.28

TOTAL AMOUNT OF PAYMENT ($) 340.00

Complete if Known	
Application Number	09/483,724
Filing Date	January 14, 2000
First Named Inventor	Sharon S. Liu
Examiner Name	AKPATI, ODAICHE T.
Group/Art Unit	2131
Attorney Docket No.	15437-0112

METHOD OF PAYMENT (check one)

1. ☒ Throughout the pendency of this application, please charge any additional fees, including any required extension of time fees, and credit all overpayments to deposit account 50-1302. A duplicate of this sheet is enclosed.

Deposit Account Number: 50-1302

Deposit Account Name: Hickman Palermo Truong & Becker, LLP

2. ☒ Payment Enclosed:
☒ Check ☐ Money Order ☐ Other

3. ☐ Applicant(s) is entitled to small entity status. See 37 CFR 1.27.

FEE CALCULATION

1. BASIC FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1001	790	2001	395	Utility filing fee	
1002	350	2002	175	Design filing fee	
1003	550	2003	275	Plant filing fee	
1004	790	2004	395	Reissue filing fee	
1005	160	2005	80	Provisional filing fee	

SUBTOTAL (1) ($) 0.00

2. EXTRA CLAIM FEES

	Highest Paid Claims		Extra Claims		Fee from Below		Fee Paid
Total Claims	%	-20**=	%	X	18.00	=	0.00
Independent Claims	%	- 3**=	%	X	86.00	=	0.00
Multiple Dependent						=	

**or number previously paid, if greater; For Reissues, see below

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description
1202	18	2202	9	Claims in excess of 20
1201	88	2201	44	Independent claims in excess of 3
1203	300	2203	150	Multiple dependent claim, if not paid
1204	88	2204	44	**Reissue independent claims over original patent
1205	18	2205	9	**Reissue claims in excess of 20 and over original patent

SUBTOTAL (2) ($) 0.00

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
1051	130	2051	65	Surcharge – late filing fee or oath	
1052	50	2052	25	Surcharge – late provisional filing fee or cover sheet.	
1053	130	1053	130	Non-English specification	
1812	2,520	1812	2,520	For filing a request for reexamination	
1804	920*	1804	920*	Requesting publication of SIR prior to Examiner action	
1805	1,840*	1805	1,840*	Requesting publication of SIR after Examiner action	
1251	110	2251	55	Extension for reply within first month	
1252	430	2252	215	Extension for reply within second month	
1253	980	2253	490	Extension for reply within third month	
1254	1,530	2254	765	Extension for reply within fourth month	
1255	2080	2255	1040	Extension for reply within fifth month	
1401	340	2401	170	Notice of Appeal	
1402	340	2402	170	Filing a brief in support of an appeal	340.00
1403	300	2403	150	Request for oral hearing	
1451	1,510	1451	1,510	Petition to institute a public use proceeding	
1452	110	2452	55	Petition to revive – unavoidable	
1453	1,370	2453	685	Petition to revive – unintentional	
1501	1,370	2501	685	Utility issue fee (or reissue)	
1502	490	2502	245	Design issue fee	
1503	660	2503	330	Plant issue fee	
1460	130	1460	130	Petitions to the Commissioner	
1807	50	1807	50	Petitions related to provisional applications	
1806	180	1806	180	Submission of information Disclosure Stmt	
8021	40	8021	40	Recording each patent assignment per property (times number of properties)	
1809	790	2809	395	Filing a submission after final rejection (37 CFR § 1.129(a))	
1810	790	2810	395	For each additional invention to be examined (37 CFR § 1.129(b))	
				Other fee (specify) _____	
				Other fee (specify) _____	

*Reduced by Basic Filing Fee Paid SUBTOTAL (3) ($) 340.00

SUBMITTED BY

Name (Print/Type)	Christian A. Nicholes	Registration No. (Attorney/Agent)	50,266	Telephone	(408) 414-1080
Signature	Christian Nicholes			Date	December 6, 2004

WARNING: **Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.**

Burden Hour Statement: This form is estimated to take 0.2 hours to complete. Time will vary depending upon the needs of the individual case. Any comments on the amount of time you are required to complete this form should be sent to the Chief Information Officer, Patent and Trademark Office, Washington, DC 20231. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Mail Stop Amend, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.